APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/765,916	01/18/2001	Frederic Canut	260/087	8270

23639   7590   08/02/2007
BINGHAM MCCUTCHEN LLP
Three Embarcadero Center
San Francisco, CA 94111-4067

EXAMINER
KANG, INSUN

ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
08/02/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Application Number: 09/765,916
Filing Date: January 18, 2001
Appellant(s): CANUT ET AL.

MAILED

AUG 0 2 2007

Technology Center 2100

Gerald Chan
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed on 4/2/2007 appealing from the Office action mailed

on 9/1/2006.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings, which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is incorrect. The applicant filed an amendment after final rejection on 11/6/2006 and the amendment has been entered.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

Pieper et al., US Pg.Pub. 2003/0005419

Cain et al., "Portable Software Library Optimization," 2/1998

Kum et al., IEEE 0-7803-5041, 3/1999

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claims 1-3, 8-16 and 21-26 are rejected under 35 U.S.C. 103(a) as being unpatentable

over Pieper et al (US 2003/0005419) in view of Cain et al ("Portable Software Library

Optimization," 2/1998) hereinafter referred to as "Cain."

Claims 4-7 and 17-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over

Pieper et al (US 2003/0005419) in view of Cain et al ("Portable Software Library Optimization,"

2/1998) hereinafter referred to as "Cain" and further in view of Kum et al. (0-7803-5041-3/99,

IEEE).

Claim Rejections - 35 USC § 103

1.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in
> section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are
> such that the subject matter as a whole would have been obvious at the time the invention was made to a person
> having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the
> manner in which the invention was made.

2.   Claims 1-3, 8-16 and 21-26 are rejected under 35 U.S.C. 103(a) as being unpatentable

over Pieper et al (US 2003/0005419) in view of Cain et al ("Portable Software Library

Optimization," 2/1998) hereinafter referred to as "Cain."

Regarding claim 1:

Pieper et al. disclose: a method of optimizing a software program for a target processor to meet

performance objectives, where the software program is coded in a high-level Language (par.

0009; par. 0018; 0020), the method comprising the steps of: (a) optimizing the software program

such that a resulting first optimized form of the software program is at least partially independent of the target processor and is at least partially coded in the high-level language, determining a first performance profile for the first optimized form of the software program, and comparing the first performance profile with the performance objectives (par. 0020; 0030; 0031); (b) based on the results of comparing the first performance profile with the performance objectives, if the performance objectives are not met by the first optimized form of the software program, then optimizing the first optimized form of the software program such that a resulting second optimized form of the software program includes at least one portion that is dependent on the target processor and is coded in the high-level language (par. 0031, 0020; par. 0045);

Pieper et al. do not explicitly disclose flagging at least one portion to indicate that the at least one portion is dependent on the target processor if the first optimized form of the software program is optimized to create the second optimized form of the software program. However, Cain teaches that using flags was known in the art of software development and optimization, at the time applicant's invention was made, to mark or identify some portions or whole code as an event of some type or having a special purpose or capability ("#include directive is used to retrieve the desired system-specific API," page 7). It would have been obvious for one having ordinary skill in the art of computer software development and optimization to modify Pieper's disclosed system to flag the modified target dependent code. The modification would be obvious because one having ordinary skill in the art would be motivated to identify the target specific code for efficient optimization and portability (page 6-7) as taught by Cain.

Regarding claim 2:

The rejection of claim 1 is incorporated, and further, Pieper et al. disclose: (bl) determining a second performance profile for the second optimized form of the software program, and comparing the second performance profile with the performance objectives (par. 0031; 0032; 0044) as claimed.

Regarding claim 3:

The rejection of claim 2 is incorporated, and further, Pieper et al. disclose:

-optimizing the second optimized form of the software program such that a resulting third optimized form of the software program is at least partially dependent on the target processor and includes portions coded in a low-level language of the target processor (par. 0031) as claimed.

Regarding claim 9:

Pieper et al. further disclose the act of implementing reference code comprises code profiling (par. 0031, 0042 ; 0046 ; 0048 ; 0049 ; 0052) as claimed.

Regarding claim 8, this claim is another version of the claimed method discussed in claim 9, wherein all claim limitations also have been addressed and/or covered in cited areas as set forth the above.

Regarding claim 10:

The rejection of claim 1 is incorporated, and further, Pieper et al. disclose :

-the act of optimization predicted to improve resulting assembly code (0031; 0032; 0009).

Regarding claim 11:

The rejection of claim 1 is incorporated, and further, Pieper et al. disclose the act of tuning

low-level functions (0031) as claimed.

Regarding claim 12:

The rejection of claim 1 is incorporated, and further, Pieper et al. disclose the act of

manual assembly optimization.   Hand-coded assembly for optimized performance is necessary

for performance critical routines such as graphics or math library routines as they often must

access low-level machine instructions for optimal execution performance.   Therefore,

accordingly, Pieper et al. anticipate this claim. See also 0009 and 0018.

Regarding claim 13:

The rejection of claim 1 is incorporated, and further, Pieper et al. the act of feature tuning (0031;

0032).

Per claims 14-16 and 21-26, they are the computer-readable medium versions of claims

1-3 and 8-13, respectively, and are rejected for the same reasons set forth in connection with the

rejection of claims 1-3 and 8-13 above.

3.   Claims 4-7 and 17-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over

Pieper et al (US 2003/0005419) in view of Cain et al ("Portable Software Library Optimization,"

2/1998) hereinafter referred to as "Cain" and further in view of Kum et al. (0-7803-5041-3/99,

IEEE).

Regarding claim 4:

The rejection of claim 1 is incorporated, and further, Pieper et al. and Cain do not explicitly teach

a floating-point implementation. However, Kum et al. disclose deriving a floating point

implementation (pg 2163, introduction, par. 3, "the ranges of floating point variables are

estimated by the simulation of the range estimation program that is automatically generated from

the original floating-point version," see also Figure 1) for the purpose of automatic scaling of all

numbers so that the numbers use the full word length available and for the purpose of reducing

the risk of overflow. Therefore, it would have been obvious to a person of ordinary skill in the

art to incorporate the teachings of Kum et al. to the system of Pieper et al and Cain. The

modification would be obvious to include the floating-point implementation because of the

automatic scaling of each number to use the full word length of the mantissa so that accurate

representation of numbers can be obtained while minimizing the risk of overflow and

quantization errors (pg 2163, introduction, par. 3).

Regarding claim 5:

The rejection of claim 1 is incorporated, and further, Pieper et al. andCain do not explicitly teach

a fixed point implementation. However, Kum et al. disclose the method of claim 1 in which step

(a) comprises the act of deriving a fixed point implementation so that "assembly coding and

manual scaling can be avoided and the translated C programs are executed very efficiently" in

fixed-point DSPs (pg 2163, introduction, lines 1-15). Therefore, it would have been obvious to a

person of ordinary skill in the art to incorporate the teachings of Kum et al. to the system of

Pieper et al and Cain. The modification would be obvious to include the fixed-point

implementation so that round-off errors can be prevented and target dependent scaling shift can

be minimized while obtaining fast real-time processing with less power and memory usage (pg

2163, introduction, lines 1-15).

Regarding claim 6:

The rejection of claim 5 is incorporated, and further, Pieper et al. and Cain do not explicitly teach

the act of processing qualification. However, Kum et al. further disclose the act of processing

qualification (Introduction, par.3; simulation-based integer word-length determination, pg 2165,

shift reduction, par. 10; pg 2163, par. 6; pg 2166, Concluding remarks) so that cost effective and

high quality fast real-time processing with less power and memory usage can be obtained while

reducing quantization noise (Introduction, par.3; simulation-based integer word-length

determination, pg 2165, shift reduction, par. 10; pg 2163, par. 6; pg 2166, Concluding remarks).

Therefore, it would have been obvious to a person of ordinary skill in the art to incorporate the

teachings of Kum et al. to the system of Pieper et al and Cain. The modification would be

obvious to include the act of processing qualification for the purpose of high quality processing

with minimized quantization noise.

Regarding claim 7:

The rejection of claim 5 is incorporated, and further,, Pieper et al. and Cain do not explicitly

teach the act of implementation sizing. However, Kum et al. further disclose the act of

implementation sizing (abstract; Introduction, pg 2163, par.3; pg 2163, simulation-based integer

word-length determination) by program-profiling results (pg 2164-2165, Sift reduction) so that

estimation of code size for the target can be obtained and the risk of overflow can be prevented.

Therefore, it would have been obvious to a person having ordinary skill in the art to incorporate

the teachings of Kum et al. to the system of Pieper et al and Cain. The modification would be

obvious to include the act of implementation sizing for the purpose of code size estimation so

that the risk of overflow can be prevented (pg 2164-2165, Sift reduction).

Per claims 17-20, they are the computer-readable medium versions of claims 4-7,

respectively, and are rejected for the same reasons set forth in connection with the rejection of

claims 4-7 above.

(10) Response to Argument

Per claims 1-3,8-16, and 21-26:

The appellant contends that:

1) Pieper does not disclose the first optimization from a resulting first optimized form of the software

program is completely independent of the target processor and is at least partially coded in the high-level language.

The term substantially independent in Pieper inherently requires that a portion or part of the optimized code of

Pieper be dependent or not independent of the architecture of the target processor. Applicants further note that the

Examiner has admitted that the term substantially and completely convey different scopes. As such, it would be

inappropriate to consider the term substantially in Pieper to be anticipatory of the limitation completely (brief, page

5).

2) In Pieper, the machine-independent code is used during a compilation process and therefore, the

machine-independent code is certainly not a result of an optimization process. This is further evidenced by the

disclosure of Pieper, which distinguishes a compilation from an optimization process. In particular, figure 2 and

paragraph 18 of Pieper disclose that a first set of computer program instructions in a high level program instruction

language 52 is converted (compiled) by compilation processes 50 into a second set of computer program instructions

in a low level program instruction language 74. This operation by Pieper is merely a compilation and is not an

optimization. As such, the machine-independent code as described in Pieper clearly cannot be considered a resulting first optimized form of the software program as recited in claim 1 (brief, page 6).

In response to the argument 1), the appellant has changed the phrase "at least partially independent" to "completely independent" in attempt to overcome Pieper. The original term was "substantially." Pieper indeed states that the code output by the optimization is in an intermediate level program code language that is "substantially independent of the architecture of the target processor (page 3, 0030)." However, the instant specification also recites that the "steps of first optimizing the software program in the high-level language, using optimizations that are substantially independent of the target processor to host the application (specification, page 2 lines 13-16)." See also appellant's argument filed on 4/16/2004. The substantially independent optimization can refer to the possible inclusion of low level/machine dependent optimization at some point. Furthermore, the word "completely independent" does not appear anywhere throughout the specification. In page 6, lines 6-9, the instant specification states that the preferred high level language is one that is completely portable between all probable DSP targets...optimization techniques particular to the language are preferably used. The appellant explains that the term "completely portable" means "completely machine independent" in the response filed on 11/6/2006 (page 6). The specification states that the "portability of the application is maintained and some optimization is integrated into the high level without using assembly language code (specification, page 11, lines 5-11). According to the specification, a target (machine) independent optimization is a generic optimization that does not include assembly or machine language code (specification, page 11, lines 5-11). Therefore, the

completely independent optimization is a generic optimization that does not include machine-specific code.

Pieper discloses a sequence of compilation procedures applied to high-level source code such as C++ (page 2, 0018; page 1, 0009). In response to the appellant's argument 2), the appellant does not acknowledge that a compilation procedure includes an optimization process (high level and low level at a different stage of compilation). One skilled in the pertinent art would know that a high-level language compiler forms intermediate code which is machine-independent and optimizes the intermediate code in the middle end of the compilation stage, while formulating the target object code, which is also optimized by using machine-specific optimizations at a later stage of the compilation. In Pieper, the compiler optimizes the intermediate code which is "machine-independent code that is first generated by the process from the input source code (page 3, 0020)" by using machine (target) independent optimizations such as loop blocking ("conventional program flow optimization techniques...conventional loop optimizations to the modified code...loop blocking," page 5, 0045). Pieper clearly states that the "first time the code generated... is analyzed" and the machine-independent intermediate code is modified so as to "optimize its program flow " and processed by a loop-rewriting process (page 5, 0045). The resulted optimized code is still machine-independent because the optimization does not include any machine dependent optimization at this point. In Pieper, the conventional optimization techniques used in this process performed in the intermediate code are generic without any low level optimization (in assembly/machine level). Like Pieper, the instant invention also uses a target independent optimization such as loop reduction that is generic for high-level languages in page 21, lines 10-14. This proves that the resulting first optimized form

in Pieper by using the generic optimizations is therefore completely target independent.

Therefore, despite appellant's numerous attempts by changing words in order to overcome

Pieper, the scope of Pieper in optimization is the same as in the instant invention.

Accordingly, the applicant's argument that Pieper does not disclose the first optimization

from a resulting first optimized form of the software program is completely independent of the

target processor and is at least partially coded in the high-level language is not persuasive.

In response to appellant's argument that Cain does not remedy the deficiencies of Pieper

because "Cain is merely relied on to reject subject matter of ...flagging ...on the target

processor...not relied upon for...the above discussed limitations (brief, page 6)," Pieper

discloses the limitations, as addressed above.

Per claims 2-3, 8-13, 15-16, and 21-26:

The applicant states that claims 2-3, 8-13, 15-16, and 21-26 are allowable as being

dependent on the independent claims 1 and 14. As shown above, Pieper in view of Cain

discloses the limitations in the independent claims 1 and 14, the argument that claims 2-3, 8-13,

15-16, and 21-26 are allowable as being dependent on the allowable base claims is moot.

Accordingly, the rejections of claims 2-3, 8-13, 15-16, and 21-26 are maintained.
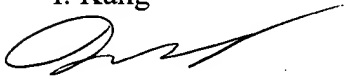
(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related

Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.
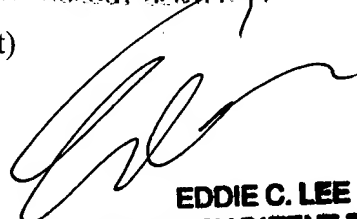
Respectfully submitted,

I. Kang

Conferees:

Meng-Ai An (2193 SPE)

/MENG-AI T. AN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Eddie Lee (Appeal Specialist)

EDDIE C. LEE
SUPERVISORY PATENT EXAMINER